题号：E

武汉理工大学第八届电工电子

创新设计大赛设计报告

题目：水温控制系统（E题）

参赛者：李锐戈

学院班级：自动化学院自动化zy1301班

联系方式：18707169086

评分标准：

|  |  |
| --- | --- |
| 项目 | 满分 |
| 设计与总结报告：方案比较，理论分析与计算，电路图及有关设计文件，测试方法与仪器，测试数据及测试结果分析 | 80 |
| 附上仿真图或实物效果图 | 20 |

[摘要 1](#_Toc17464)

[1系统方案 1](#_Toc23275)

[1.1系统设计 1](#_Toc974)

[1.1.1系统原理分析 1](#_Toc31583)

[1.1.2系统结构框图 1](#_Toc21102)

[1.2方案选择 1](#_Toc5807)

[1.2.1处理器方案 1](#_Toc9619)

[1.2.2温度传感器方案 2](#_Toc32285)

[1.2.3控制方案 2](#_Toc23625)

[1.2.4加热方案 2](#_Toc25621)

[2软件设计 3](#_Toc6321)

[2.1模糊控制算法 3](#_Toc15885)

[2.1.1温度偏差（Terror）隶属度函数 3](#_Toc20186)

[2.1.2温度差值变化率（Dif）隶属度函数 3](#_Toc31363)

[2.1.3输出功率（OUT）隶属度函数 3](#_Toc31562)

[2.1.4模糊控制规则的建立 4](#_Toc21466)

[2.1.5模糊结果的计算 4](#_Toc18156)

[2.2软件流程图 5](#_Toc3872)

[3硬件设计 5](#_Toc19257)

[3.1 处理器 5](#_Toc11780)

[3.2加热电路 5](#_Toc210)

[3.3温度传感器 5](#_Toc5512)

[3.4显示模块 6](#_Toc29684)

[4系统仿真测试 6](#_Toc30514)

[5参数调试 6](#_Toc30051)

[6结果分析 7](#_Toc30916)

[参考文献 7](#_Toc29623)

[附录 8](#_Toc14025)

摘要

根据题目要求，采用K60处理器，利用温度传感器DS18B20，实时测量水的温度，根据模糊控制的策略，通过控制继电器的通断，来调整加热棒的输出功率，达到控制目标温度的目的。

关键字：K60处理器 温度传感器 模糊控制

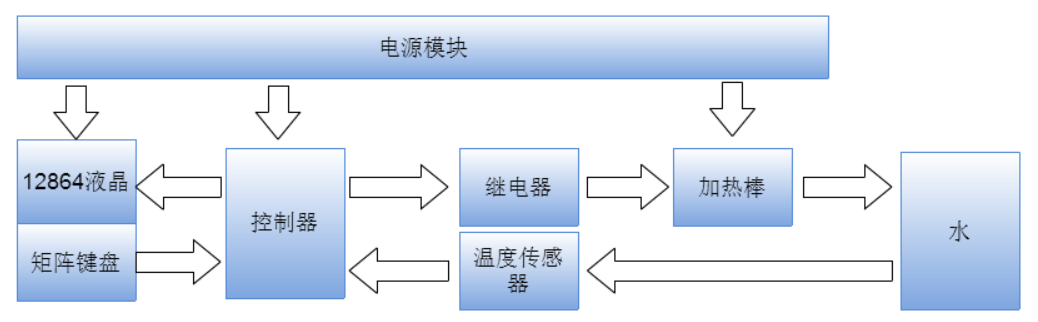
# 1系统方案

## 1.1系统设计

### 1.1.1系统原理分析

根据题目要求，通过温度传感器实时监测当前温度数据，与给定目标值进行比较，当温度降低，或给予风扇降温等扰动，或给予阶跃的给定目标值时，根据当前与给定的差值和当前温度差值的变化率，利用模糊控制算法，控制加热棒的输出功率。从而使水温能够稳定或者跟随给定目标。

### 1.1.2系统结构框图



## 1.2方案选择

### 1.2.1处理器方案

方案一：采用51单片机，这种单片机成本极低，最小系统电路简单，程序容易编写。但存在运算能力不足，外设不足等缺点。

方案二：采用K60处理器。这种ARM架构的处理器电路和程序比较复杂，成本较高。但运算速度快，外设丰富，在系统设计时有较大的灵活性。

通过比较分析，认为采用方案二较为合适。

### 1.2.2温度传感器方案

方案一：采用TI公司的LM19传感器，该传感器供电电压为+2.4V到+5.5V，量程范围-55℃到+130℃，采用模拟输出，因此需要使用ADC采集，采集温度可以根据输出电压可由技术手册公式计算得到。

方案二：采用TI公司的LM135传感器，该传感器的测量范围为−55℃到150℃，采用模拟输出，需要搭建外围检测电路，并可以使用外围电路校准输出温度。测量误差为0.3℃。

方案三：采用DS18B20，该传感器是数字输出，测量范围-55℃到125℃，可采用寄生电源工作方式而不需要外部电源供电，因此只需要一条数字信号线和一条地线即可完成测量。测量精度为0.0625℃，转换时间较长且由转换精度决定。

通过比较分析，决定采用方案三即DS18B20的方案，该方案硬件电路简单，测量精度高。虽然转换时间较长，但由于本系统时间常数大，传感器转换时间对控制影响不大。

### 1.2.3控制方案

方案一：采用PID控制，比例积分微分控制是一种非常通用的控制算法，对于线性定常系统有比较好的控制效果。它不需要系统的模型，参数一般可以通过现场整定得到。其缺点也比较明显，即对较复杂的系统控制效果比较差。性能指标较难达到。

方案二：采用离线生成模糊控制表，在线查表的控制方式。模糊控制可以模拟人工控制的过程。可以根据专家经验，大大改善系统性能，提高响应速度和超调。通过matlab离线建立模糊控制的规则表和隶属度函数，并导出为控制表。将控制表导入到控制系统中，在控制时，根据输入查表即可得到输出。该方案对处理器的运算速度要求很小，但控制规则表和隶属度函数的整定较为麻烦。

方案三：采用在线计算的模糊控制。相对于离线生成，在线查表的模糊控制方法。该方案则是直接将模糊控制的运算写入软件代码，进行实时运算和控制。通过上位机，可以实时修改模糊规则和隶属度函数，方便调试，但处理器的运算量将会非常大。

根据题目要求，通过对比分析，模糊控制才能获得更好的控制效果，为了方便调试，并且考虑到K60的优秀的运算性能，决定采用方案三。

### 1.2.4加热方案

方案一：采用小功率直流电热棒，通过开关调压电路控制输入电压来控制输出的功率。

方案二：采用大功率交流电热棒，通过交流调功控制输出功率。

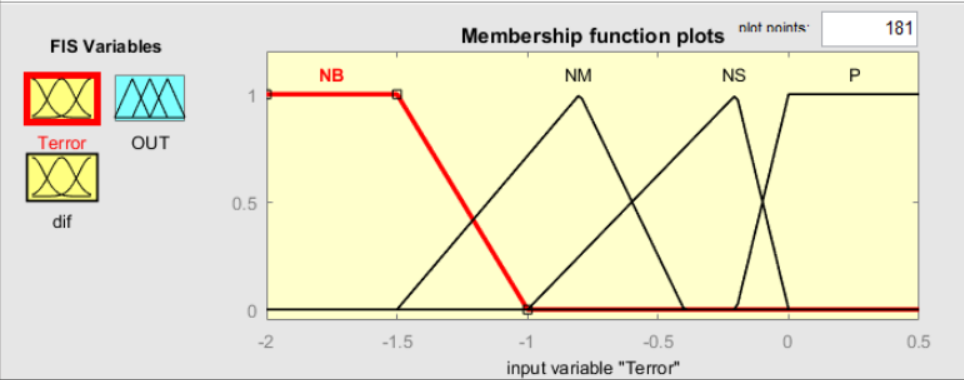
 考虑到水的比热容为4.2×10³ J/(kg℃)，即每升高1℃需要4200J热量。在不考虑热量散失及效率100%的理想条件下，从40℃升至60℃需要84000J，需要500W的电热棒加热2.8分钟。因此需要使用比较大功率的加热设备才能满足题目的要求，并考虑到系统的时间常数比较大，选择方案二。

# 2软件设计

## 2.1模糊控制算法

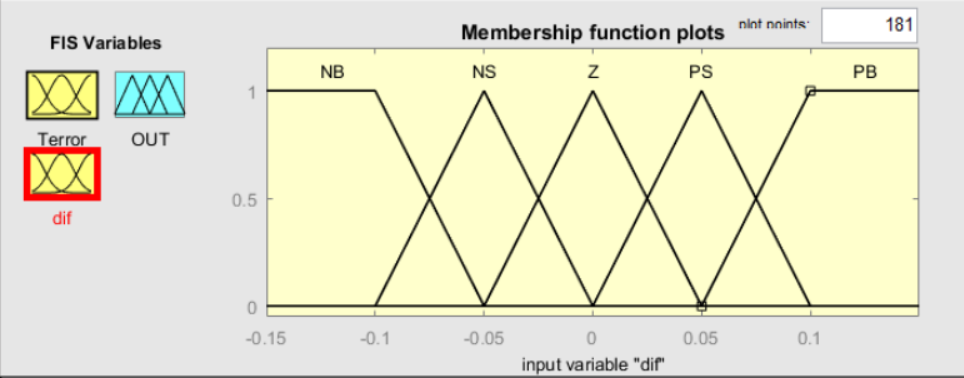
### 2.1.1温度偏差（Terror）隶属度函数

初步建立温度偏差隶属度函数如图所示。当前水温和目标水温的差值分为四级，负大（NB），负中（NM），负小（NS），正（P），将输入温度偏差模糊化。



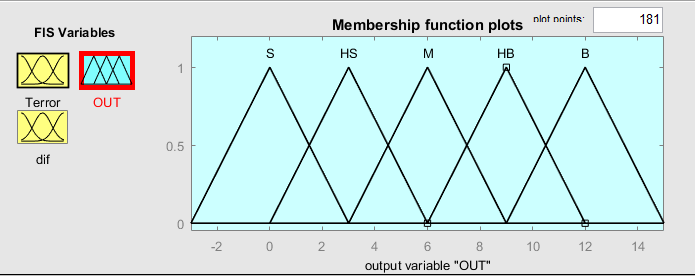
### 2.1.2温度差值变化率（Dif）隶属度函数

初步建立本次采样水温较上次水温的差值隶属度函数如图所示，共分五级：负大（NB）,负小（NS），零（Z），正小（PS）,正大（PB），将温度差值变化率模糊化。



### 2.1.3输出功率（OUT）隶属度函数

初步建立输出给加热器的功率隶属度函数如图所示。共分五级，分别为小（S），半小（HS）中（M），半大（HB），大（B）。其中最小输出为0，最大为12。

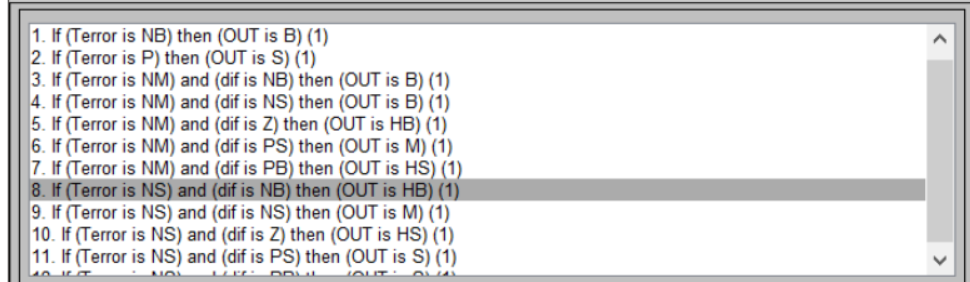


### 2.1.4模糊控制规则的建立

水温控制系统的时间常数较大，因此在水温与设定值相差比较大时（Terror = NB），加热器应该饱和输出（OUT = B），以提高响应速度。当水温高于设置时（Terror = P），加热器应该使用小输出甚至停止输出（OUT = S）以抑制超调。

当水温低于设定值但差距不大时，应该根据偏差和偏差变化率控制加热器的输出，提高系统的指标。

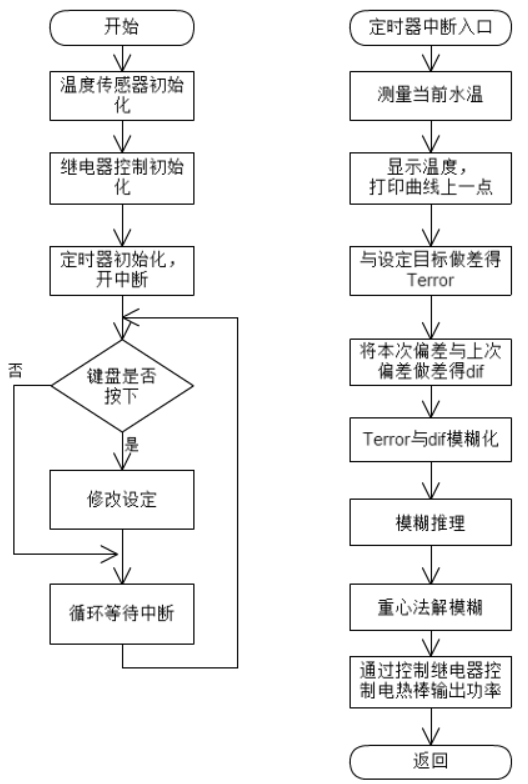
由此初步建立模糊控制规则表如图所示。



### 2.1.5模糊结果的计算

当模糊推理结束后需要进行反模糊处理，输出需要的结果，这里采用比较常见的重心法解模糊。通过计算重心点所在位置来得出模糊结果。

## 2.2软件流程图



# 3硬件设计

## 3.1 处理器

使用自制的K60最小系统核心板，其自带稳压芯片，有防止过流过压及反接电路。自带蓝牙接口，LED指示灯等，方便调试。原理图见附录。

## 3.2加热电路

用带光耦隔离的继电器控制加热棒，给继电器输入低频的PWM信号，利用开关原理进行交流调功，控制加热棒的输出功率。

## 3.3温度传感器

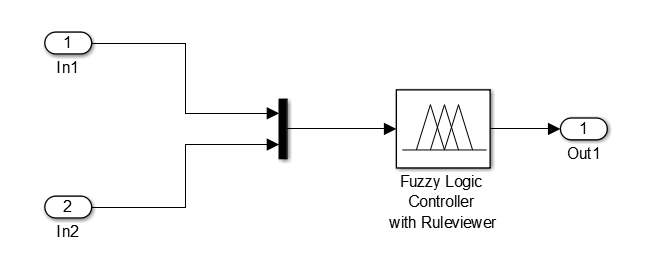
使用一个带防水套的DS18B20数字温度传感器，将其VCC和GND同时接地，信号线接IO口。使其工作在寄生电源模式。利用IO口模拟通信时序，读出传感器所测得的值。

## 3.4显示模块

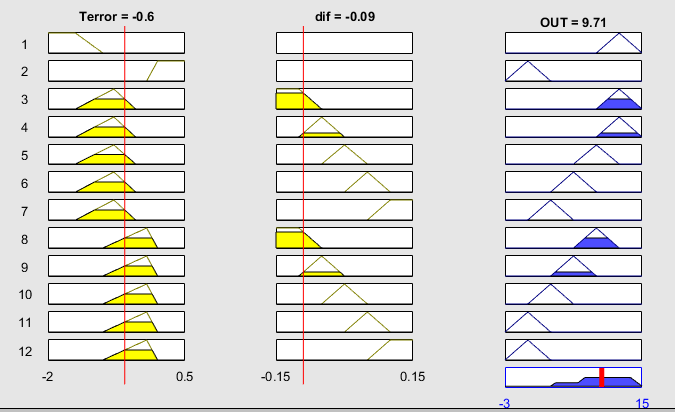
题目要求实时打印温度曲线，因此加入液晶显示模块，使用12806液晶，与处理器进行SPI通信，控制简单可靠。

# 4系统仿真测试

在simulink中建立模糊控制仿真模型如下



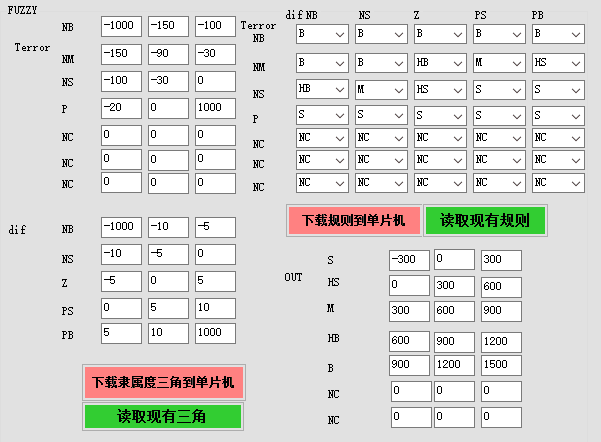
取一些典型的输入，在systemtest中测试其输出。观察输出是否合理。



从测试结果可以看出，输出是符合预期的。

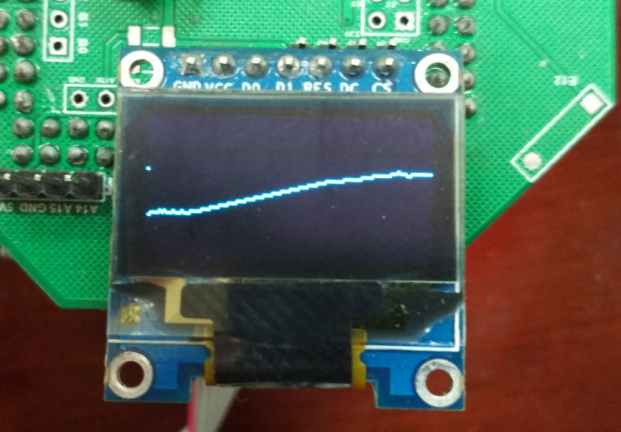
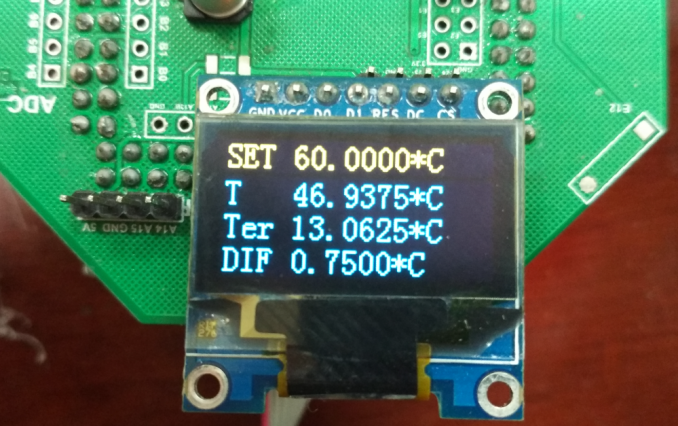
# 5参数调试

利用之前为模糊控制调试而设计的上位机。可在线实时微调隶属度函数和模糊规则表。根据经验调试，使系统达到稳态误差小，动态响应快，超调小的题目要求。



# 6结果分析

由于系统采用了模糊控制的算法，因此系统性能极佳，具有良好的动态特性和静态特性。温度控制的静态误差小于0.2℃。当设定温度突变时，系统调节时间和超调量很小。加上12864液晶的配合，可以实时显示温度和自动打印水温随时间变化的曲线。



# 参考文献

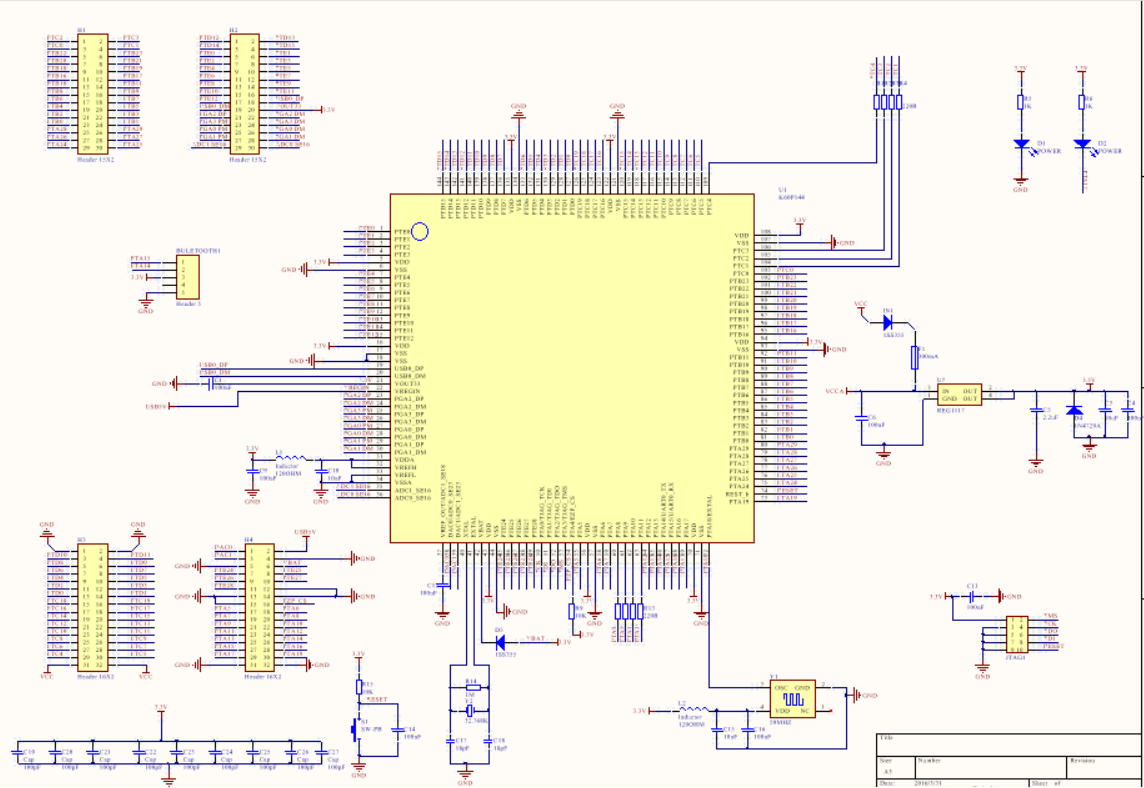
[1]胡寿松主编 自动控制原理（第四版）科学出版社，2001

[2]王兆安 刘进军主编 电力电子技术（第五版）机械工业出版社 2015

[3]张晓华主编 控制系统数字仿真与CAD（第3版）机械工业出版社 2016

# 附录

主控板原理图：



模糊控制代码节选：

#define T\_NB 0 #define T\_NM 1 #define T\_NS 2 #define T\_P 3

#define D\_NB 0 #define D\_NS 1 #define D\_Z 2 #define D\_PS 3

#define D\_PB 4 #define O\_NB 0 #define O\_HB 1 #define O\_M 2

#define O\_HS 3 #define O\_S 4

uint8 rule\_list[4][5] =

{ //D\_NB D\_NS D\_Z D\_PS D\_PB

{O\_B, O\_B, O\_B, O\_B, O\_B}, /\* T\_NB \*/

{O\_B, O\_B, O\_HB, O\_M, O\_HS }, /\* T\_NM \*/

{O\_HB, O\_M, O\_HS, O\_S, O\_S}, /\* T\_NS \*/

{O\_S, O\_S, O\_S, O\_S, O\_S} /\* T\_P \*/

};

float T\_memf[4][3] =

{

{-10, -1.5, -0.1}, //T\_NB

{-1.5, -0.9, -0.3}, //T\_NM

{-0.1, -0.3, 0}, //T\_NS

{-0.2, 0, 10} //T\_P

};

float D\_memf[5][3]=

{

{-10, -0.1, -0.05}, //D\_NB

{-0.1, -0.05, 0}, //D\_NS

{-0.05, 0, 0.05}, //D\_Z

{0, 0.05, 0.1}, //D\_PS

{0.05, 0.1, 10}, //D\_PB

};

float O\_memf[5][3] =

{

{-3., 0., 3.}, //O\_B

{0, 3., 6.}, //O\_HB

{3., 6., 9.}, //O\_M

{6., 9., 12.}, //O\_HS

{9., 12., 15.} //O\_S

};

//对当前输入求所有输入隶属度三角形的面积并遍历规则表进行模糊推理得出输出隶属度三角

for ( i = 0 ;i < 4; i++ ) //Terror

{

for ( j = 0;j < 5; j++ ) //Dif

{

if ( E\_DATA > T\_memf[i][0] && E\_DATA < T\_memf[i][2]

&& EC\_DATA > D\_memf[j][0] && EC\_DATA < D\_memf[j][2] )

{

if ( E\_DATA <= T\_memf[i][1] )

{

e\_temp = (1./(T\_memf[i][1]-T\_memf[i][0])) \* (E\_DATA - T\_memf[i][0]);

}

else

{

e\_temp = (1./(T\_memf[i][1]-T\_memf[i][2])) \* (E\_DATA - T\_memf[i][1])+1;

}

if ( EC\_DATA < D\_memf[j][1] )

{

ec\_temp = 1.0/(D\_memf[j][1]-D\_memf[j][0]) \* (EC\_DATA - D\_memf[j][0]);

}

else

{

ec\_temp = (1./(D\_memf[j][1]-D\_memf[j][2]))\* (EC\_DATA - D\_memf[j][1])+1;

}

//遍历规则表

result[rule\_list[i][j]] += e\_temp \* ec\_temp;

}

}

}

//反模糊，重心法

u\_temp += result[0]\*1\* ( ( (O\_memf[0+1][0] -O\_memf[0][2] )/ 2 + O\_memf[0][2] ) - O\_memf[0][0]) \* O\_memf[0][1];

u2\_temp += result[0]\* ( ( (O\_memf[0+1][0] - O\_memf[0][2] )/ 2 + O\_memf[0][2] ) - O\_memf[0][0]);

for(i=1;i<4;i++)

{

u\_temp += result[i]\*1\*( ( (O\_memf[i+1][0] -O\_memf[i][2] )/ 2 + O\_memf[i][2] )

- (( O\_memf[i][0] - O\_memf[i-1][2])/2+ O\_memf[i-1][2]) ) \* O\_memf[i][1];

u2\_temp += result[i]\*1\*( ( (O\_memf[i+1][0] -O\_memf[i][2] )/ 2 + O\_memf[i][2] )

- (( O\_memf[i][0] - O\_memf[i-1][2])/2+ O\_memf[i-1][2]) );

}

u\_temp += result[4]\*1\* ( O\_memf[4][2] - ((O\_memf[4][0] - O\_memf[3][2])/2 + O\_memf[3][2]) ) \* O\_memf[4][1];

u2\_temp += result[4]\*1\* ( O\_memf[4][2] - ((O\_memf[4][0] - O\_memf[3][2])/2 + O\_memf[3][2]) );

output = (u\_temp / u2\_temp);

return (output);

}

系统部分照片：

